project_final

December 18, 2019

[1]:
```
import numpy as np
from gurobipy import *
```

[2]:
```
demand = [

[
    [0,0,0,0,0],[100,200,100,400,300],[50,50,50,50,50]
],

[
    [25,25,25,25,25],[0,0,0,0,0],[25,25,25,25,25]
],

[
    [40,40,40,40,40],[400,200,300,200,400],[0,0,0,0,0]
],

]

c_dict = {'A':0, 'B':1, 'C':2}
i_dict = {0:'A', 1:'B',  2:'C'}



repostion_cost = [
    [
        0,7,3
    ],
    [
        7,0,6
    ],
    [
        3,6,0
    ]
]

c_hold = 10
```

```
total_fleet = 1200
```

[27]:
```
model = Model("Final_Project_v1")
x = [[[0]*5 for i in range(3)] for j in range (3)] #number of cargo deliver at␣
 ↪day t
y = [[[0]*5 for i in range(3)] for j in range (3)] #number of plane␣
 ↪repositioning at day t
z = [[[0]*5 for i in range(3)] for j in range (3)] #number of cargo holding at␣
 ↪day t
```

[28]:
```
#OBJ
objExpr = LinExpr()
for T in range(5):
    for i in range(3):
        for j in range(3):
            repo_fleet = model.addVar(vtype = GRB.INTEGER,
                                      name =␣
 ↪'y_'+str(i_dict[i])+'_'+str(i_dict[j])+'_day_'+str(T))
            y[i][j][T] = repo_fleet
            objExpr += repostion_cost[i][j]*repo_fleet
            if i != j:
                hold_var = model.addVar(vtype = GRB.INTEGER,
                                        name =␣
 ↪'z_'+str(i_dict[i])+'_'+str(i_dict[j]+'_day_'+str(T)))
                z[i][j][T] = hold_var
                objExpr += c_hold*hold_var
                deliver_cargo = model.addVar(vtype = GRB.INTEGER,
                                        name = 'x_'+str(i_dict[i])+
                                          '_'+str(i_dict[j]+'_day_'+str(T)))
                x[i][j][T] = deliver_cargo
model.update()
model.setObjective(objExpr,GRB.MINIMIZE)
```

[29]:
```
#constrain on aircraft flow
for T in range(5):
    for i in range(3):
        constExpr_l = LinExpr()
        constExpr_r = LinExpr()
        for j in range(3):
            prev_day = 4 if T == 0 else (T - 1)
            constExpr_l += y[j][i][prev_day] + x[j][i][prev_day]
            constExpr_r += y[i][j][T] + x[i][j][T]
        model.addConstr(lhs = constExpr_l, sense = GRB.EQUAL, rhs =␣
 ↪constExpr_r,name = 'aircraft'+str(T)+str(i_dict[i]))
model.update()
```

[30]:
```
#constraints on cargos
for T in range(5):
    for i in range(3):
        for j in range(3):
            if i != j:
                prev_day = 4 if T == 0 else (T - 1)
                constExpr_l = LinExpr()
                constExpr_r = LinExpr()
                constExpr_l += demand[i][j][T] + z[i][j][prev_day]
                constExpr_r += x[i][j][T] + z[i][j][T]
                model.addConstr(lhs = constExpr_l, sense = GRB.EQUAL, rhs =␣
 ↪constExpr_r,name = 'cargo_loads'+str(T)+str(i_dict[i])+str(i_dict[j]))
model.update()
```

[31]:
```
#we have to satisfy the total cargo each week
for i in range(3):
    for j in range(3):
        if i != j:
            constExpr_l = LinExpr()
            for T in range(5):
                constExpr_l += x[i][j][T]
            model.addConstr(lhs = constExpr_l, sense = GRB.EQUAL, rhs =␣
 ↪sum(demand[i][j]),name = 'total_deliver'+str(i_dict[i])+str(i_dict[j]))
model.update()
```

[32]:
```
#constraint on total number of plane
for T in range(5):
    constExpr_l = LinExpr()
    for i in range(3):
        for j in range(3):
            constExpr_l += x[i][j][T] + y[i][j][T]
    model.addConstr(lhs = constExpr_l, sense = GRB.EQUAL, rhs = 1200 ,name =␣
 ↪'total_plane'+str(T))
model.update()
```

[33]:
```
model.optimize()
```

```
Optimize a model with 56 rows, 105 columns and 345 nonzeros
Variable types: 0 continuous, 105 integer (0 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [3e+00, 1e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [2e+01, 2e+03]
```

```
Presolve time: 0.00s
Presolved: 56 rows, 105 columns, 345 nonzeros
Variable types: 0 continuous, 105 integer (0 binary)

Root relaxation: objective 1.792500e+04, 30 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

*    0     0               0    17925.000000 17925.0000  0.00%     -    0s

Explored 0 nodes (30 simplex iterations) in 0.01 seconds
Thread count was 8 (of 8 available processors)

Solution count 1: 17925

Optimal solution found (tolerance 1.00e-04)
Best objective 1.792500000000e+04, best bound 1.792500000000e+04, gap 0.0000%
```

[34]:
```python
print("optimal Objective: \n" + str(model.ObjVal))
```

```
optimal Objective:
17925.0
```

[35]:
```python
x
```

[35]:
```
[[[0, 0, 0, 0, 0],
  [<gurobi.Var x_A_B_day_0 (value 290.0)>,
   <gurobi.Var x_A_B_day_1 (value 200.0)>,
   <gurobi.Var x_A_B_day_2 (value 100.0)>,
   <gurobi.Var x_A_B_day_3 (value 400.0)>,
   <gurobi.Var x_A_B_day_4 (value 110.0)>],
  [<gurobi.Var x_A_C_day_0 (value 50.0)>,
   <gurobi.Var x_A_C_day_1 (value 50.0)>,
   <gurobi.Var x_A_C_day_2 (value 50.0)>,
   <gurobi.Var x_A_C_day_3 (value 50.0)>,
   <gurobi.Var x_A_C_day_4 (value 50.0)>]],
 [[<gurobi.Var x_B_A_day_0 (value 25.0)>,
   <gurobi.Var x_B_A_day_1 (value 25.0)>,
   <gurobi.Var x_B_A_day_2 (value 25.0)>,
   <gurobi.Var x_B_A_day_3 (value 25.0)>,
   <gurobi.Var x_B_A_day_4 (value 25.0)>],
  [0, 0, 0, 0, 0],
  [<gurobi.Var x_B_C_day_0 (value 25.0)>,
   <gurobi.Var x_B_C_day_1 (value 25.0)>,
   <gurobi.Var x_B_C_day_2 (value 25.0)>,
   <gurobi.Var x_B_C_day_3 (value 25.0)>,
   <gurobi.Var x_B_C_day_4 (value 25.0)>]],
 [[<gurobi.Var x_C_A_day_0 (value 20.0)>,
   <gurobi.Var x_C_A_day_1 (value 60.0)>,
   <gurobi.Var x_C_A_day_2 (value 40.0)>,
   <gurobi.Var x_C_A_day_3 (value 40.0)>,
   <gurobi.Var x_C_A_day_4 (value 40.0)>],
  [<gurobi.Var x_C_B_day_0 (value 330.0)>,
   <gurobi.Var x_C_B_day_1 (value 270.0)>,
   <gurobi.Var x_C_B_day_2 (value 300.0)>,
   <gurobi.Var x_C_B_day_3 (value 200.0)>,
   <gurobi.Var x_C_B_day_4 (value 400.0)>],
  [0, 0, 0, 0, 0]]]
```

[36]:
```
y
```

[36]:
```
[[[<gurobi.Var y_A_A_day_0 (value -0.0)>,
   <gurobi.Var y_A_A_day_1 (value -0.0)>,
   <gurobi.Var y_A_A_day_2 (value 75.0)>,
   <gurobi.Var y_A_A_day_3 (value -0.0)>,
   <gurobi.Var y_A_A_day_4 (value -0.0)>],
  [<gurobi.Var y_A_B_day_0 (value -0.0)>,
   <gurobi.Var y_A_B_day_1 (value -0.0)>,
   <gurobi.Var y_A_B_day_2 (value -0.0)>,
   <gurobi.Var y_A_B_day_3 (value -0.0)>,
   <gurobi.Var y_A_B_day_4 (value -0.0)>],
  [<gurobi.Var y_A_C_day_0 (value -0.0)>,
   <gurobi.Var y_A_C_day_1 (value -0.0)>,
   <gurobi.Var y_A_C_day_2 (value -0.0)>,
   <gurobi.Var y_A_C_day_3 (value -0.0)>,
   <gurobi.Var y_A_C_day_4 (value -0.0)>]],
 [[<gurobi.Var y_B_A_day_0 (value 205.0)>,
   <gurobi.Var y_B_A_day_1 (value 140.0)>,
   <gurobi.Var y_B_A_day_2 (value 310.0)>,
   <gurobi.Var y_B_A_day_3 (value 95.0)>,
   <gurobi.Var y_B_A_day_4 (value 275.0)>],
  [<gurobi.Var y_B_B_day_0 (value -0.0)>,
   <gurobi.Var y_B_B_day_1 (value -0.0)>,
   <gurobi.Var y_B_B_day_2 (value 110.0)>,
   <gurobi.Var y_B_B_day_3 (value -0.0)>,
   <gurobi.Var y_B_B_day_4 (value -0.0)>],
  [<gurobi.Var y_B_C_day_0 (value 255.0)>,
   <gurobi.Var y_B_C_day_1 (value 430.0)>,
   <gurobi.Var y_B_C_day_2 (value -0.0)>,
   <gurobi.Var y_B_C_day_3 (value 365.0)>,
   <gurobi.Var y_B_C_day_4 (value 275.0)>]],
 [[<gurobi.Var y_C_A_day_0 (value -0.0)>,
   <gurobi.Var y_C_A_day_1 (value -0.0)>,
   <gurobi.Var y_C_A_day_2 (value -0.0)>,
   <gurobi.Var y_C_A_day_3 (value -0.0)>,
  <gurobi.Var y_C_A_day_4 (value -0.0)>],
 [<gurobi.Var y_C_B_day_0 (value -0.0)>,
  <gurobi.Var y_C_B_day_1 (value -0.0)>,
  <gurobi.Var y_C_B_day_2 (value -0.0)>,
  <gurobi.Var y_C_B_day_3 (value -0.0)>,
  <gurobi.Var y_C_B_day_4 (value -0.0)>],
 [<gurobi.Var y_C_C_day_0 (value -0.0)>,
  <gurobi.Var y_C_C_day_1 (value -0.0)>,
  <gurobi.Var y_C_C_day_2 (value 165.0)>,
  <gurobi.Var y_C_C_day_3 (value -0.0)>,
  <gurobi.Var y_C_C_day_4 (value -0.0)>]]]
```

[37]:
```
z
```

[37]:
```
[[[0, 0, 0, 0, 0],
  [<gurobi.Var z_A_B_day_0 (value -0.0)>,
   <gurobi.Var z_A_B_day_1 (value -0.0)>,
   <gurobi.Var z_A_B_day_2 (value -0.0)>,
   <gurobi.Var z_A_B_day_3 (value -0.0)>,
   <gurobi.Var z_A_B_day_4 (value 190.0)>],
  [<gurobi.Var z_A_C_day_0 (value -0.0)>,
   <gurobi.Var z_A_C_day_1 (value -0.0)>,
   <gurobi.Var z_A_C_day_2 (value -0.0)>,
   <gurobi.Var z_A_C_day_3 (value -0.0)>,
   <gurobi.Var z_A_C_day_4 (value -0.0)>]],
 [[<gurobi.Var z_B_A_day_0 (value -0.0)>,
   <gurobi.Var z_B_A_day_1 (value -0.0)>,
   <gurobi.Var z_B_A_day_2 (value -0.0)>,
   <gurobi.Var z_B_A_day_3 (value -0.0)>,
   <gurobi.Var z_B_A_day_4 (value -0.0)>],
  [0, 0, 0, 0, 0],
  [<gurobi.Var z_B_C_day_0 (value -0.0)>,
   <gurobi.Var z_B_C_day_1 (value -0.0)>,
   <gurobi.Var z_B_C_day_2 (value -0.0)>,
   <gurobi.Var z_B_C_day_3 (value -0.0)>,
   <gurobi.Var z_B_C_day_4 (value -0.0)>]],
 [[<gurobi.Var z_C_A_day_0 (value 20.0)>,
   <gurobi.Var z_C_A_day_1 (value -0.0)>,
   <gurobi.Var z_C_A_day_2 (value -0.0)>,
   <gurobi.Var z_C_A_day_3 (value -0.0)>,
   <gurobi.Var z_C_A_day_4 (value -0.0)>],
  [<gurobi.Var z_C_B_day_0 (value 70.0)>,
   <gurobi.Var z_C_B_day_1 (value -0.0)>,
   <gurobi.Var z_C_B_day_2 (value -0.0)>,
   <gurobi.Var z_C_B_day_3 (value -0.0)>,
   <gurobi.Var z_C_B_day_4 (value -0.0)>],
  [0, 0, 0, 0, 0]]]
```

[39]:
```
model.write(filename = "final_project_v1.lp")
```

[ ]:
```

```

[ ]:
```

```